****

Report: hw4

Author: F74045018 廖其忻<cayon.1318.96@hotmail.com>

Class: 甲班

Description:

In this homework, I have used 2 types of functions which are quicksort and split to do the rearrangement of the number. I first generate N number of integer or floating number , then store into array and call the function to do the rearrangement with descending order.The types of number, which is integer of floating number, can be controlled by argument list.

Code:

```
 11 #include<stdio.h>
 12 #include<stdlib.h>
 13 #include<time.h>
 14
 15 void quicksort1(int a1[],int low1,int high1)
//function used to rearrange INTEGER
 16 {
 17    int middle1;
 18    if (low1>=high1) return;
 19    middle1=split1(a1,low1,high1);
 20    quicksort1(a1,low1,middle1-1);
 21    quicksort1(a1,middle1+1,high1);
 22 }
 23
 24
 25 int split1(int a1[],int low1,int high1) //sub-function
used to rearrange INTEGER
 26 {
 27    int part_element1 = a1[low1];
 28    for(;;)
 29    {
 30       while (low1<high1 && part_element1<=a1[high1])
 31       high1--;
```

```
32        if(low1>=high1) break;
33        a1[low1++]=a1[high1];
34
35        while(low1<high1 && a1[low1]<=part_element1)
36        low1++;
37        if (low1>=high1) break;
38        a1[high1--]=a1[low1];
39    }
40
41    a1[high1]=part_element1;
42    return high1;
43 }
44
45
46 void quicksort2(float a2[],int low2,int high2)
//function used to rearrange FLOATING POINT NUMBER
47 {
48    int middle2;
49    if (low2>=high2) return;
50    middle2=split2(a2,low2,high2);
51    quicksort2(a2,low2,middle2-1);
52    quicksort2(a2,middle2+1,high2);
53 }
54
55
56 int split2(float a2[],int low2,int high2)
//sub-function used to rearrange FLOATING POINT NUMBER
57 {
58    int part_element2 = a2[low2];
59     for(;;)
60     {
61        while (low2<high2 && part_element2<=a2[high2])
62        high2--;
63        if(low2>=high2) break;
64        a2[low2++]=a2[high2];
65
66        while(low2<high2 && a2[low2]<=part_element2)
67        low2++;
```

```
 68         if (low2>=high2) break;
 69         a2[high2--]=a2[low2];
 70     }
 71
 72    a2[high2]=part_element2;
 73    return high2;
 74  }
 75
 76
 77 int main(int argc, char *argv[])
 78 {
 79    int N,T;    //N determines how many number to be
generated; T determines the type of number to be generated
 80    N=atoi(argv[1]);
 81    T=atoi(argv[2]);
 82
 83    if(T==0)    //when the type of number to be generated
is INTEGER
 84    {
 85        printf("There are %d integer number generated and
arranged in descending order\n",N);
 86
 87        int a[10000]={0};   //initialize the array a
 88        srand((unsigned int)time(NULL));
 89
 90        int i;  //generate N integer randomly and store
into array a
 91        for(i=0;i<=N;i++)
 92        {
 93           a[i]=rand()%10001;  //the number generated is
with 0~10000
 94        }
 95
 96        quicksort1(a,0,N-1);    //call function
quicksort1
 97
 98        printf("in sorted descending order: \n");
 99
```

```
100        int k;  //as the function in TB rearrange in
ascending order. so if want to print out the number with
descending order, the array have to be print out rever    sely.
101            for(k=N-1;k>=0;k--)
102                printf("%d ",a[k]);
103        printf("\n");
104    }
105
106
107
108    if(T==1)    //when the type of number to be generated
is FLOATING POINT NUMBER
109    {
110        printf("There are %d floating point number
generated and arranged in descending order\n",N);
111
112        float b[10000]={0}; //initialize the array b
113        srand((unsigned int)time(NULL));
114
115        int j;  //generate N floating point number and
store into array b
116        float lowest_range=0;
117        float highest_range=10000;
118        for(j=0;j<=N;j++)
119        {
120
b[j]=(highest_range-lowest_range)*rand()/(RAND_MAX)-lowes
t_range;   //the way of generate randomly the floating point
number within 0~10000
121        }
122    quicksort2(b,0,N-1);
123
124        printf("in sorted descending order: \n");
125
126        int k;  //as the function in TB rearrange in
ascending order. so if want to print out the number with
descending order, the array have to be print out rever
sely.
```

```
127           for(k=N-1;k>=0;k--)
128              printf("%.4f ",b[k]);
129        printf("\n");
130     }
131
132 return 0;   //the program ends.
133 }
```

Compilation:

```
gcc -o hw4 hw4.c
```

Execution:

```
./hw4 N T
```

Output:

```
F74045018@c-2015-1:~/hw4> ./hw4 7 1
There are 7 floating point number generated and arranged in
descending order
in sorted descending order:
 9510.0000 8799.8262 8234.0000 5588.0000 3337.0000 2272.2229
808.0000
F74045018@c-2015-1:~/hw4> ./hw4 9 0
There are 9 integer number generated and arranged in
descending order
in sorted descending order:
9542 9164 8511 6447 5606 4987 3628 3091 43
```

Error message:

```
      hw4.c: In function ‘main’:
hw4.c:110:10: error: ‘highest_range’ undeclared (first use
in this function)

b[j]=(highest_range-lowest_range)*rand()/(RAND_MAX)-low;
         ^
hw4.c:110:10: note: each undeclared identifier is reported
only once for e
ach function it appears in
```